AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim 1 (currently amended): A ~~memory management method for~~ method of managing a memory of a ~~testing~~ device for testing a data storage device, said method comprising:

reading a data block from said data storage device into a block of memory, said memory block comprising a plurality of smaller memory chunks;

for each said memory chunk, maintaining in real time a record of whether any data stored in said memory chunk is required to be maintained for use by a test program; and

under a condition of said records indicating that the data in all [[of]] the memory chunks of said data block need not be maintained for use by said test program, deleting said data block.

Claim 2 (currently amended): The method as claimed in claim 1, wherein maintaining a record comprises:

for each said memory chunk, maintaining [[a]] flag data indicating [[a]] the status of data ~~contained within~~ in said memory chunk.

Claim 3 (currently amended): The method as claimed in claim 1, wherein maintaining [[a]] the record[[,]] comprises:

maintaining a count of ~~individual~~ said memory chunks[[,]] for which the data stored in said memory chunks is required to be maintained.

Claim 4 (currently amended): The method as claimed in claim 1, wherein maintaining [[a]] the record, comprises:

maintaining a count of individual said memory chunks for which data stored in [[said]] memory chunks is required to be maintained; and

maintaining, for each memory chunk, a pointer to a memory block of which said memory chunk forms a part.

Claim 5 (currently amended): The method as claimed in claim 1, wherein maintaining [[a]] the record[[,]] comprises:

maintaining for each memory chunk, a pointer to a memory block of which said memory chunk forms a part; and

maintaining a pointer to a data chunk stored in said memory chunk.

Claim 6 (original): The method as claimed in claim 1, further comprising:

generating a message to delete said data block in response to a signal from said test program that no further testing of said data block is required.

Claim 7 (currently amended): A ~~data management~~ method [[for]] of managing ~~a plurality of~~ plural data blocks in a memory device for testing of [[said]] data in the blocks by using at least one test program, said method comprising:

receiving a plurality of said data blocks in said memory device, each said data block ~~comprising a plurality of~~ including plural data chunks;

setting a plurality of flags ~~for indicating~~ that indicate whether each of said data blocks [[are]] is to be maintained or not maintained for reading by said at least one test program;

maintaining ~~individual~~ said data blocks having a ~~corresponding said~~ set flag indicating that said data block is to be maintained; and

deleting said data blocks having flags indicating that said data blocks are not to be maintained.

Claim 8 (currently amended): The method as described in claim 7, wherein each of said data block comprises at least one data chunk, said method comprising:

for each said data chunk, maintaining a pointer to a data block from which said data chunk originates.

Claim 9 (original): The method as claimed in claim 7, comprising maintaining a flag indicating that a reader application has finished processing a data block.

Claim 10 (currently amended): The method as claimed in claim 7, wherein each of said data block comprises at least one data chunk, said method comprising maintaining a record of [[a]] the number of data chunks which are currently in use for [[a]] one of said data [[block]] blocks.

Claim 11 (currently amended): The method as claimed in claim 7, wherein each of said data block comprises at least one data chunk, said method comprising:

maintaining a counter record indicating [[a]] the number of data chunks in use for [[a]] one of said data [[block]] blocks.

Claim 12 (currently amended): The method as claimed in claim 7, comprising:

maintaining [[a]] flag data indicating whether or not a reader application has finished data processing a data block.

Claim 13 (currently amended): A method of managing a memory for maintaining a plurality of data blocks in a memory device, such that said data blocks are made available to at least one reader device which reads data from said data blocks for processing by at least one test component , said method comprising:

reading [[a]] one of said data [[block]] blocks into said memory;

dividing said one data block into a plurality of data chunks;

creating a corresponding respective flag for each of said data [[chunk]] chunks of said data block;

initialising said data flag to an "in use" status;

selecting individual data chunks of said data block for reading by said reader device;

reading a block pointer of [[a]] one of the selected [[said]] data [[chunk]] chunks, said block pointer pointing to said data block;

processing [[a]] said one selected data chunk by using said at least one test component; and

applying a flag setting to said data block from which said one selected data chunk originates, said flag setting indicating that said data chunk has been processed.

Claim 14 (currently amended): The method as claimed in claim 13, further comprising:

for each of a plurality of data chunks of said data block, maintaining a corresponding respective data flag indicating whether said data chunk is in use or not;

in response to all of said data flags of said data block ~~adapting~~ having a "not in use" status, deleting said data block from said memory.

Claim 15 (currently amended): The method as claimed in claim 13, further comprising:

determining whether any further testing of said data block is required; and

[[If]] if no further testing of said data block is required, deleting all of said [[whole]] data block.

Claim 16 (currently amended): A reader component for reading a plurality of data chunks from a memory , said reader component comprising respective ~~sub components~~ sub-components for:

creating flags for a plurality of data chunks, said flags being arranged for indicating whether each of said data [[chunk]] chunks is in use or not in use;

maintaining a data block flag, said data block flag being arranged for indicating whether the said data block is in use or not in use;

determining whether said reader device has finished reading from [[a]] said data block; and

generating a signal for ~~deleting~~ indicating a data block from which said reader component has finished reading [[from]] is to be deleted.

Claim 17 (currently amended): The reader component as claimed in claim 16, further comprising a ~~sub-component~~ sub-component for generating a message to ~~delete~~ indicating that a data block from said memory is to be deleted.

Claim 18 (currently amended): A memory ~~management means~~ manager for managing a plurality of data blocks in a memory , said memory management means comprising:

means for ~~effecting receipt of~~ receiving a plurality of said data blocks in said memory , each said data block comprising a plurality of data chunks;

means for setting a plurality of data block flags[[,]] indicating whether each of said data blocks are in use or not in use by at least one program; and

means for determining whether said data blocks are to be maintained in said memory or not, depending on [[a]] the status of a ~~corresponding said~~ flag indicating that said data block is in use, or is not in use by said at least one program.

Claim 19 (currently amended): The memory ~~management means~~ manager as claimed in claim 18, comprising:

means for maintaining a plurality of data pointers for indicating, for each of a plurality of data chunks including plural ~~comprising a plurality of~~ data blocks, a data block from which each of said data [[chunk]] chunks originates.

Claim 20 (currently amended): The memory ~~management means~~ manager as claimed in claim 18, comprising:

means for maintaining a set of flags indicating that a reader application has finished processing [[a]] one of said data [[block]] blocks.

Claim 21 (currently amended): The memory ~~management means~~ manager as claimed in claim 18, comprising:

means for maintaining a record of [[a]] the number of data chunks which are currently in use [[for a]] in one of said data [[block]] blocks.

Claim 22 (currently amended): The memory ~~management means~~ manager as claimed in claim 18, comprising:

means for maintaining [[a]] flag data indicating whether or not a reader application has finished data processing of a data block.

Claim 23 (currently amended): The memory ~~management means~~ manager as claimed in claim 18, further comprising:

means for generating a message to delete ~~a whole~~ said data block from said memory[[,]] if no further testing of said data block is required.

Claim 24 (currently amended): A method of ~~memory management for~~ managing a plurality of memory blocks in a memory device for testing data in said memory blocks, said method comprising:

partitioning a plurality of said memory blocks in said memory device, each of said memory ~~block comprising a plurality of~~ block having plural memory chunks each adapted for storing a corresponding respective data chunk;

setting a plurality of flags ~~for indicating~~ that indicate whether data stored in each of said memory blocks are to be maintained or not maintained; and maintaining data stored in individual ones of said memory blocks having a ~~corresponding said~~ flag indicating that said data of said memory block is to be maintained.

Claim 25 (currently amended): A ~~data management~~ method [[for]] of managing ~~a plurality of~~ plural data blocks in a memory device, said method comprising:

creating a memory block ~~comprising a plurality of~~ having plural memory chunks;

storing a block of data in said memory block, such that individual data chunks ~~comprising~~ include in said data block are stored in said ~~plurality of~~ plural memory chunks;

maintaining a record of [[a]] the number of active data chunks in said memory block; and

under conditions where [[a said]] the active number of data chunks in a memory block is zero, deleting all of said data blocks from said memory block.

Claim 26 (currently amended): A memory management method comprising:

reading a data block into a block of memory, said memory block comprising a plurality of smaller memory chunks;

for each said memory chunk, maintaining a record of whether data stored in said memory chunk is required to be maintained;

under a condition of the record indicating that data in all [[of]] the memory chunks need not be maintained, deleting said data block.

Claim 27 (new): The reader component of claim 17, further comprising responding to the message by deleting from said memory the data block from which said reader component has finished reading from said memory.

Claim 28 (new): Apparatus for performing the method of claim 1.

Claim 29 (new): Apparatus for performing the method of claim 13.

Claim 30 (new): Apparatus for performing the method of claim 24.

Claim 31 (new): Apparatus for performing the method of claim 25.

Claim 32 (new): Apparatus for performing the method of claim 26.